
2.10 KNOWLEDGE BASE

In a mission-level combat model, decisions are based on the decision-maker's collection of knowledge which may be acquired through training, doctrine, and experience as well as other avenues. This part of the knowledge base contains tactical knowledge, and tactics are typically represented in a series of "if ..., then ..." logical constructs. Tactics tells the decision-maker what decision to make or action to perform (once values, thresholds, or conditions within the tactical situation are satisfied.) For example, part of the submarine weapons officer's tactical knowledge base includes "if the target bears 90° at a range of 1,000 yards, then issue the torpedo fire order." Decision-making systems acquire situational knowledge as the exercise unfolds. Situational knowledge tells the decision-maker when to think or act. At the start of the simulation, the weapons officer knows that he will issue the fire torpedo order when the target is at a specific relative bearing and range. However, the weapons officer does not know when, or even if, the criteria will be satisfied. The decision-maker will issue the fire order once the tactical situation satisfies the tactical criteria.

Tactical knowledge within SWEG consists of tactics and orders. In general, tactics are rules or operating procedures, and the user can define tactics for each type player which has at least one thinker system. This tactical knowledge includes evaluation rates, maneuver profiles, resource allocation categories, and contingency plans. The completeness of a player's tactical knowledge base symbolizes the quality of its training and preparedness. The creation of tactics has a well-defined syntax which guides the user through building the tactical knowledge base. The user-defined portions of the tactical knowledge base are inputs to the react and ponder processes.

Orders are specific instructions for a scenario player. SWEG allows the user to define orders that differentiate the tactical operation of each player instance including different instantiations of the same player type. For example, all the fast attack submarines (SSNs) have the same thinking systems, thinking activities and servicing times; however, the implementation of these activities can be differentiated by giving each specific SSN a different engagement zone. Orders can also include other geographical controls such as phase lines or points for the start of maneuvers. If a player type has alternate plans or tactics defined, orders can establish the initial plan or tactic used by a specific player instance.

Situational knowledge consists of iconic images, a list of things to think about, and perceptions. Iconic images are the result of fleeting physical stimuli that are presented to the mental processor. A decision-maker may be somewhat aware of the stimuli before actually forming a perception. Examples include half-listening to a speaker while concentrating on something else or seeing something in the blink of an eye. The only iconic images in SWEG are scanning sensor results that are not noticed before the next scan.

In SWEG, the list of things to think about is termed the pending queue. It consists of the future mental processes that have been set up for a player by initial orders or perceptions, a physical event, or a previous mental event. The operation of the pending queue is discussed in depth in Section 2.11 Decision-Making Logic Processes.

In SWEG, perceptions are the knowledge that the player possesses about itself and about other players which basically includes everything in a scenario. In the real world, a person can have perceptions about people and things that actually exist as well as perceptions

about people or things that possibly exist. The perceptions about possibly existing people and things could include intuited information such as pattern recognition and spurious perceptions such as those in which clutter and noise are misperceived as actual people or things. SWEG does not simulate perceptions about possible entities; all perceptions in a SWEG scenario are based on the actual players defined within the scenario.

Perceptions about another player can include the following types of information: position and orientation, movement, status information, communication network affiliations, status of requests made and received, and command and control hierarchies. Perceptions of self are implicit, contain all this information, and are always accurate.

Perceptions of other players within a SWEG scenario may be incomplete which means they may not contain all possible information and they may be inaccurate. These perceptions may derive from both implicit and explicit information sources. In the real world, differences between real data and perceived data can be introduced by physical processes such as sensing or communicating. They can also be introduced by perceptual recognition, assimilation and extrapolation, and mistakes in the mental processes. In SWEG, all sensor-derived perceptions are initially accurate, and they become inaccurate only when they are out-of-date.

In the real world, there may be no connection between the amount of detail in reality and the detail in any perception. Simulated reality has a certain resolution of detail which is defined by the user or hard-coded into the model. Simulated perceptions could not reasonably have more detail than this, but they can have less detail. Simulated perceptions should be able to differ from simulated reality in terms of detail or aggregation, quantity of information, quality of information, and timeliness; however, in SWEG they may differ only in timeliness.

Perceptions and all other types of knowledge possessed by a player may be kept in long-term, medium-term, or short-term memory. These three categories refer to the willingness of the player to discard the information. Implicit perceptions, tactics, and orders normally belong in long-term memory; a player is not willing to discard them because they are not volatile and/or they cannot be replaced. Medium-term memory consists of sensor-derived perceptions. A player would be more willing to discard these because they may be out-of-date and they can be obtained again. Iconic images are in short-term memory.

2.10.1 Functional Element Design Requirements

- a. SWEG will provide for friendly player perceptions through the creation of command chains and the KNOWS data item.
- b. SWEG will provide the user with the capability to build a player's tactical knowledge base through the TACTICS data item. This data item can include specifications for evaluation rates; movement options, maneuver profiles, and formation definitions; resource allocation; and contingency plans.

2.10.2 Functional Element Design Approach

Friendly Players

An important SWEG concept directly related to perceptions is the friendly player. A friendly player is defined to be a player whose perception does not require explicit sensor detection. Some friendly players are implicitly perceived such as the immediate commanders and subordinates on explicitly-defined command chains to which the perceiving player belongs. However, only one perception of each commander or subordinate will exist regardless of the number of command chains to which they mutually belong.

Other friendly players can be perceived through their designation in the KNOWS data item in the Scenario Data Base. If the user includes the HAS clause, the player will also perceive the friendly player's materiel status. The KNOWS data item is frequently used when the user wants a player to communicate with another player without using the normal command chain structure or when that structure would be awkward.

Although a player may be perceived as friendly, no assumption exists in SWEG that the friendly player will be, or should be, treated in any manner that could be characterized as friendly. Also, no assumption can be made that the perceived player will act in a friendly manner. Also, incorrect information may be provided should the user want the player to misperceive the location of a friendly player. SWEG does permit a player to implicitly update its locational information about its perceived friendly players. However, not all information about the perceived player is available since the perception is held at the player level of detail. Specifically, if the perceived player has multiple platforms at different locations, the entire player is perceived to be at the location of its first-defined platform. Last, a player can receive orders and requests only from a friendly player.

All other perceptions are sensor-derived. These players are perceived at the platform level of detail. Sensor-derived perceptions may be directly obtained from a player's own sensors, or they may be indirectly obtained via a communication message from another player. The data available from these perceptions is determined by the sensor receiver which collected the data. The only way for one player to know the location of another player's multiple platforms is through direct or indirect sensor-derived perceptions.

Tactical Knowledge Base

The user creates a tactical knowledge base for each player that will engage in thinking activities during an exercise. The user employs SWEG's scenario conflict language (SCL) to define evaluation rates; movement options, maneuver profiles, and formation definitions; resource allocation; and contingency plans in the TDB TACTIC data item. The specific syntax for each part of the tactical knowledge base acts as scaffolding which guides the user in creating each player's unique tactics. The user-provided information from the TDB is parsed to create a scenario player's tactical knowledge. During runtime, the tactical and situational knowledge is interwoven, and each scenario player's knowledge base is created. Resource allocation and contingency plans are the most complicated portions of the tactical knowledge base, and their design approach is discussed in detail.

Resource Allocation

A SWEG player must have user-defined resource allocation tactics in order to react. The purpose of the resource allocation knowledge base is to provide SWEG with sufficient information at runtime so it can effect the player allocating the most satisfactory of the available resources to the perceived target. The user creates resource allocation tactics within the RESOURCE-ALLOCATION data item. The data contained in this data item are allocation options, target options, resource options, tactical criteria, selection criteria, and filters.

The allocation option is the building block for resource allocation. It contains the filtering and selection criteria that a player will use in choosing the resources to allocate to or deallocate from a perceived target. SWEG currently implements 23 allocation options which are extensions of the nine resource allocation categories. When determining which allocation options to define for a player, the user must consider both the overall scenario being simulated and each player's role within the scenario. The user defines those allocation options which match the player's real world reactive activities. SWEG does not limit the number of allocation options which the user can define. Table 2.10-1 lists the resource allocation categories and options as well as a short description.

TABLE 2.10-1. Resource Allocation Categories and Options.

Category	Option	Description
Absorb	absorb-start	defines conditions under which resources will be absorbed
Comm Method Selection	comm-method-selection	defines conditions under which a particular net will be selected to send a particular message
Emission Control	emcon	defines conditions under which resources will be turned on or off
Intell	intell-send	defines conditions under which messages about a perceived target will be sent
Lethal Assignment	lethal-assignment-queue-add	defines criteria for placing a perceived target on a queue of potential candidates for assignment
	lethal-assignment-queue-drop	defines criteria for removing a perceived target from a queue of potential candidates for assignment
	lethal-assignment-start	defines conditions under which a commander will make assignments to subordinates
	lethal-assignment-stop	defines conditions under which a commander will cancel assignments
Lethal Engagement	lethal-engage-queue-add	defines criteria for placing a perceived target on a queue of potential candidates for engagement
	lethal-engage-queue-drop	defines criteria for removing a perceived target from a queue of potential candidates
	lethal-engage-start	defines conditions under which a player will initiate a lethal engagement of a perceived target
	lethal-engage-stop	defines conditions under which a player will stop an engagement

TABLE 2.10-1. Resource Allocation Categories and Options. (Contd.)

Category	Option	Description
Lethal Engagement (Contd.)	lethal-engage-firing-start	defines conditions under which a player will stop an engagement
	lethal-engage-firing-stop	defines conditions under which a player will terminate a firing sequence during an engagement
Maneuver	mnvr-queue-add	defines criteria for placing a perceived target on a queue for potential candidates for maneuvering
	mnvr-queue-drop	defines criteria for removing a perceived target on a queue of potential candidates for maneuvering
	mnvr-start	defines criteria for determining that a platform starts maneuvering towards the target
	mnvr-stop	defines criteria for determining that a platform stops maneuvering towards the target
Nonlethal Engagement	jammer-queue-add	defines criteria for placing a perceived target on a queue of potential candidates for jamming
	jammer-queue-drop	defines criteria for removing a perceived target from a queue of potential candidates for jamming
	jammer-spot-add	defines criteria for determining that a jammer should focus power on the frequency of the emitter and try to jam it
	jammer-spot-drop	defines criteria by which a jammer should stop emitting energy on a frequency
Request	fill-request	defines conditions under which information will be requested about a perceived target

Within each allocation option, the user will provide information about the targets which the player will consider valid, the resources which the player has the ability to allocate, and the selection criteria the player will use in determining which resources are acceptable as well as choosing the specific resources to allocate.

A perceived target is determined to be valid if its type matches the type specified by the user in the allocation option's target option. The syntax for the target option includes one of three pre-defined target types and its key phrase. The target types currently implemented are TGT-TYPE, MSG-TYPE, and PERCEPTION-TYPE. The target type and key phrase which the user can include in the data item are circumscribed by the resource allocation category, and Table 2.10-2 lists the categories and their associated target options.

TABLE 2.10-2. Resource Allocation Categories and Target Options.

Category	Target Option	Key Phrase
Absorb	tgt-type	player, ANYONE, ALL-OTHERS, EXCEPT
Comm Method Selection	msg-type	message, ANYONE, ALL-OTHERS, EXCEPT
Emission Control	tgt-type	anyone

TABLE 2.10-2. Resource Allocation Categories and Target Options. (Contd.)

Category	Target Option	Key Phrase
Intell	perception-type	player, ANYONE, ALL-OTHERS, EXCEPT
Lethal Assignment	tgt-type	player, ANYONE, ALL-OTHERS, EXCEPT
Lethal Engagement	tgt-type	player, ANYONE, ALL-OTHERS, EXCEPT
Maneuver	tgt-type	player, ANYONE, ALL-OTHERS, EXCEPT
Nonlethal Engagement	tgt-type	communications transmitter, sensor transmitter, ANYONE, ALL-OTHERS, EXCEPT
Request	tgt-type	player, ANYONE, ALL-OTHERS, EXCEPT

The key phrase allows the user to specify a large or small set of valid targets. If the key phrase is ANYONE and the perceived target's type matches the allocation option's target type, the player will consider the perceived target as valid for resource allocation. For example, if the allocation option is JAMMER-SPOT-ADD, the key phrase is ANYONE, and the perceived target is a sensor transmitter, the player can allocate resources to the perceived target because sensor transmitter is the correct target type for this allocation option. However, if the perceived target is a bomber, the player can not allocate resources since the bomber is not a valid target type. A player key phrase contains the user-defined names of one or more type players. Similarly, a comm-xmtr or snr-xmtr key phrase contains the user-defined names of one or more transmitter systems defined in the TDB. A message key phrase refers to a specific message defined in the TDB's MESSAGE-DEFINITION data item. Except for the EMCON allocation option, the ANYONE and ALL-OTHERS key phrase can accept an EXCEPT clause.

An example from the unclassified Obruty war scenario of the airborne commander's target options for the LETHAL-ASSIGNMENT-START allocation option may be helpful in understanding how the user can author key phrases. The huge_bomber, attacker, and sar_drone are player type defined in the TDB.

```
lethal-assignment-start assign_1
    tgt-type huge_bomber
    :
    tgt-type attacker
    :
    tgt-type all-others except sar_drone
    :
end lethal-assignment-start
```

In the above example, the airborne commander employs a different set of tactics for each target type. The user can list multiple names in a key phrase if the player will employ the same tactics against all the listed target types. For example, a huge_bomber uses the same tactics to attack either a long-range or medium-range SAM site. This target options could be defined as:

```
LETHAL-ENGAGE-START engage_1
    TGT-TYPE long_sam med_sam
    :
    TGT-TYPE ALL-OTHERS
    :
end lethal-assignment-start
```

For each target option, the user specifies the types of resources that the player will attempt to allocate. A resource option contains a resource type followed by a list of resources which will be either players, platforms, or systems defined in the TDB or communication networks defined in the SDB. The contents of the resource option are circumscribed by the resource allocation category in a manner similar to the target option. Table 2.10-3 summarizes the resource options.

TABLE 2.10-3. Resource Allocation Categories and Resource Options.

Category	Resource Type	Resource Name
Absorb	thinker-type	thinker system
Comm Method Selection	comm-method	network
Emission Control	snr-tx-type	sensor transmitter system
Intell	player-type	player
Lethal Assignment	sub-type	player
Lethal Engagement	wpn-type	weapon system
Maneuver	platform-type sub-type	platform player
Nonlethal Engagement	jammer-type	disrupter system
Request	thinker-type comm-method player-type sub-type jammer-type wpn-type platform-type	player

In terms of syntax, the user builds filters for each target option. The filters contain resource options, tactical criteria, and selection criteria. Filters are convenient mechanisms for combining resource options and criteria into modular chunks of data. They also allow the use of tactical criteria to progressively refine the set of candidate resources. The allocated resources are selected from the subsets created by the filtering process. Tactical criteria express conditions that must be met by some or all of the resources, target and/or player if the resource is to become a member of the filter's subset. These conditions can be based on geometric criteria, status criteria, or other characteristics; 86 tactical criteria are currently available for use within the resource allocation filters. In some cases, the use of a specific tactical criterion is circumscribed by the allocation option.

The airborne commander's lethal-assignment-start allocation option for the TGT-TYPE huge_bomber will be used to clarify filters, tactical criteria, and the selection of resources.

```

LETHAL-ASSIGNMENT-START assign_2
  TGT-TYPE huge_bomber
    USE INPUT FOR FILTER 1
      SUB-TYPE vis_fighter
        TOTAL-ASGS NO-MORE-THAN 3 (TGTS)
        AND 2D-DIST < 350.0E3 (M)
        AND 2D-DIST > 15.0E3 (M)
      SUB-TYPE bvr_fighter
        TOTAL-ASGS NO-MORE-THAN 4 (TGTS)

```

```

        AND 2D-DIST < 350.0E3 (M)
        AND 2D-DIST > 20.0E3 M
    USE FILTER 1 SELECTIONS FOR FILTER 2
        SUB-TYPE vis_fighter
        TOTAL-ASGS NO-MORE-THAN 1 (TGTS)
        AND 2D-DIST < 200.0E3 (M)
        SUB-TYPE bvr_fighter
        TOTAL-ASGS NO-MORE-THAN 2 (TGTS)
        AND 2D-DIST < 225.0E3 (M)
    FROM FILTER 2 SELECTIONS
        CHOOSE-FROM
            bvr_fighter
            SEND-MESSAGE weapon_assignment
            REFER-TO THIS-TARGET RECIPIENT: THIS-PLAYER
            PICK-AT-MOST 2 NOW 4 TOTAL SUB-TOTAL: 3
    FROM FILTER 2 SELECTIONS
        CHOOSE-FROM
            vis_fighter
            SEND-MESSAGE weapon_assignment
            REFER-TO THIS-TARGET RECIPIENT: THIS-PLAYER
            PICK-AT-MOST 2 NOW 4 TOTAL SUB-TOTAL: 2
    FROM FILTER 1 SELECTIONS
        CHOOSE-FROM
            vis_fighter
            SEND-MESSAGE weapon_assignment
            REFER-TO THIS-TARGET RECIPIENT: THIS-PLAYER
            bvr_fighter
            SEND-MESSAGE weapon_assignment
            REFER-TO THIS-TARGET RECIPIENT: THIS-PLAYER
            PICK-AT-MOST 1 NOW 3 TOTAL SUB-TOTAL: 3
    END LETHAL-ASSIGNMENT-START

```

At the start of a target option's filtering process, an initial set of candidate resources is created. As the tactical criteria specific to a filter are evaluated, subsets of the initial candidate resource set are created, and each filter has an associated subset. In SWEG, the subset can be a replica of the original set, a true subset, or the null set. The USE data item accepts either INPUT or FILTER as its argument. From the resource options specified under the USE INPUT clause, SWEG builds the initial set of candidate resources. At runtime, the resources selected for allocation are chosen from the members of the filter subsets. The order in which the resources are selected is based on the order of the FROM FILTER entries, and the number of resources selected is determined by the PICK-AT-MOST entry if sufficient resources are found in the filter's subset. If multiple resources meet the criteria, the user cannot affect which resource is chosen. The above example will be explained in detail to demonstrate how SWEG allocates resources to the perceived target.

The first step in resource allocation is to determine the perceived target's validity. Let's assume that the abn_cmdr has a abn_cmdr_thk system defined within its player-structure which means that the abn_cmdr has a pending queue. The abn_cmdr also has a EVAL-ASG/CANCEL entry in its TIME-TO-THINK table. At a point during the exercise, the next entry on the pending queue is LETHAL-ASSIGNMENT-START, and the perceived target type for this event is huge_bomber. In this instance, the type for the perceived target matches the target type specified by the user in the target option which means that resource allocation will proceed if any huge_bombers are already on the assignment queue. If the perceived

target's type for the pending queue entry had been `big_bomber`, the `LETHAL-ASSIGNMENT-START` event would not proceed since the perceived target's type is invalid.

The second step in resource allocation is to create the initial set of candidate resources. In this example, the set of candidate resources will contain all the `vis_fighters` and `bvr_fighters` which the `abn_cmdr` perceives as friendly.

The third step evaluates the candidate resources against the tactical criteria specified in each filter. Numerous calculations are made in this step for the target, the player, and each of the candidate resources. First the target's velocity and position are calculated, and the player's velocity and position are updated. The remainder of this step is the evaluation of each candidate resource against each filter's set of tactical criteria. Each resource is evaluated in turn on a filter-by-filter basis starting with the set of candidates implied by the `USE INPUT` entry. Other calculations may be done depending on the type of resource option. For example, if the resource type is `SUB-TYPE`, the resource's current position and velocity are determined. Next, the resource's value for each of the tactical criteria within the current filter is calculated and compared against the value specified by the user. If all of a particular filter's criteria are satisfied when the resource is evaluated, the resource becomes a member of that filter's subset of candidate resources. This subset can be the input to another filter depending on how the user specified the filters in the target option. Also, the resources which are selected for allocation are chosen from the filter subsets created during the evaluation step.

After the `huge_bomber` and `abn_cmdr`'s position and velocity are determined, each candidate resource is evaluated. Since the resource type in our example is `SUB-TYPE`, the resource's location and velocity are calculated. Next, the resource's value for each of the tactical criteria is determined. The `TOTAL-ASGS` tactical criterion uses the length of the resource's current assignment list plus one, and the `2D-DIST` tactical criterion uses the planar distance from the target to the resource. These two values are resolved for each for each `vis_` or `bvr_fighter` which the `abn_cmdr` perceives as friendly. The values are then evaluated against the tactical criteria. If the resource is a `vis_fighter`, the `TOTAL-ASGS 2D-DIST` is between 15,000 and 350,000 meters or the resource is a `bvr_fighter`, the `TOTAL-ASGS 2D-DIST` is between 20,000 and 350,000 meters, the resource is a member of Filter 1's subset of candidate resources. The resources in Filter 1's subset become the input candidate resources for Filter 2. The process of calculating each resource's value for the tactical criteria and evaluating the resource's value against the user specified value is repeated for Filter 2. Those Filter 1 candidate resources which meet both of Filter 2's tactical criteria become members of Filter 2's subset of candidate resources.

The last step in resource allocation is to provide the selection criteria which will define the order and quantity of resources allocated from the target option's subsets of candidate resources. This selection order is based upon the filter subsets created in the `USE` entry and the order of the `FROM FILTER` entries. The order in which the filters are cited in the `FROM FILTER` entry determines the order in which SWEG will attempt to allocate resources. Within each `FROM FILTER` entry, one or more resource types will be specified. These resource types may have a qualifier which directs how the resource is to be used, and the qualifiers can impact the resource selection process. They can be used in any order, but the order in which they are specified is the order in which they are processed. The qualifier can be a `WITH` phrase, `REMEMBER` or `MOVEMENT-ORDER`. The allocation option circumscribes

the contents of a WITH phrase. No requirement exists to name a filter previously used in the evaluation step, and a filter can be mentioned more than once in selection criteria.

The PICK-AT-MOST entry specifies the number of resources to allocate during execution of the current allocation option as well as the total number of resources that can be allocated to the target. It is possible that fewer than the specified number of resources will be allocated if a filter subset does not contain sufficient members or it is the null set. This number of resources allocated is ultimately based on runtime considerations; allocation is constrained by how many resources the player has, and deallocation is constrained by how many are active at the time.

The NOW option specifies the number of resources within the specified resource types to select for the current execution of the allocation option. The TOTAL option specifies the maximum number of overall resources allocated to the target at any one time. A TOTAL option should only be specified for the nonlethal engagement, lethal engagement, maneuver, intell, and lethal assignment allocation categories. If the user specifies a total amount for any other category, the value will be ignored. The SUB-TOTAL option specifies the maximum number of resources of the specific resource types that can be allocated to the target.

In our example, the *bvr_fighter* candidate resources within the Filter 2 subset will be allocated first. The user has specified that no more than 2 *bvr_fighters* from the Filter 2 subset are to be allocated during the execution of this allocation option. The TOTAL option specifies that SWEG will not allocate more than 4 resources to the target, and the SUB-TOTAL option specifies that no more than 3 of the total allocated resources can be *bvr_fighters*. If the total and subtotal values are not exceeded and the Filter 2 subset contains any *bvr_fighters*, no more than 2 *bvr_fighters* will be allocated to the target.

Next, the *vis_fighters* within the Filter 2 subset will be selected if the TOTAL/SUB-TOTAL values have not been exceeded. To continue with the example, 2 *vis_fighters* will be allocated if the Filter 2 subset contains at least 2 *vis_fighters*. Again, either 0 or 1 *vis_fighter* can be allocated depending on Filter 2 subset membership.

If the total and sub-total quantities have still not been exceeded, members of the Filter 1 subset will be selected for allocation. In our example, either one *vis_fighter* or one *bvr_fighter* will be selected. No more than 3 resources can be allocated to the target when the Filter 1 subset is being processed, and no more than 3 of the total resources allocated to the target can be a *vis_* and *bvr_fighter*. SWEG permits the user to specify different TOTAL and SUB-TOTAL values in each CHOOSE-FROM clause within one allocation option.

Contingency Plans

A SWEG player must have user-defined contingency plans in order to ponder. This portion of the tactical knowledge base is organized around categories of plans, and each plan category, if used, will contain one or more user-authored plans. Plans can be developed for resources, for targets, for a mix of resources and targets, and for no targets or resources. Three syntax phrases are the building blocks for a plan: looping, alternation (or branching), and execution. At least one of the three phrases should be used once within a contingency plan; otherwise the plan will not evaluate anything or cause anything to happen. Each phrase can be used as many times as desired in a plan within certain constraints imposed

by the code implementation limits, and it can be employed in any order with the other phrases.

SWEG currently has nine designated plan categories: CMD/CONTROL, ASSIGNMENT, LAUNCH, LETHAL-ENGAGE, MOVEMENT, NON-LETHAL-ENGAGE, ABSORB, REQUEST and SEND-STATUS. The names of the plan categories have no intrinsic meaning; they simply provide convenient categories for organizing a player's contingency plans. Within each plan category, SWEG permits only one active plan at a time for a given player. Each active plan can interact with active plans in the other categories in terms of changing the active plan or causing or canceling reactive thinking that might normally be under the purview of a different category plan. As opposed to resource allocation, contingency planning does not have the same type of formal organization of categories, filters, options, and criteria. Since the names of the nine types of contingency plans are completely arbitrary, each plan can be designed to do whatever the user deems fit. This includes the intentional design of plans that have internal conflicts among the various plans. SWEG does not have any embedded logic to attempt to eliminate internal conflicts.

The number of plan instances within a plan category is not restricted in SWEG. However, one plan should be mentioned in the SDB player data item in order for contingency plans of that category to be implemented. A plan should not be thought of as a monolithic object that is implemented once or thought about once. A plan is better thought of as an attitude that has an ongoing effect on the player's thought processes. Just as attitudes or opinions can change over time, so plans can change.

The syntax of the looping, alternation, and execution phrases demonstrates the user's flexibility in authoring the ponder portion of the tactical knowledge base. The purpose of the looping phrase is to sequentially evaluate perceptions. The perceptions can represent potential targets or resources. In this respect, ponder is similar to react in that it can view objects as resources and targets. It differs from react in that a target or a resource is necessary. Targets are more generalized in ponder. For example, SECTIONS, which are a geographical concept, can be thought of as targets to which resources can be allocated for physical occupation or movement control.

The alternation phrase provides a branching mechanism within a contingency plan. The branching is determined by meeting the specified tactical criterion. The subsequent action can be another alternation phrase, an execution phrase, or a looping phrase. The alternation phrase can have multiple branches and a default, using keywords such as a single WHEN, zero or more BUT-WHEN, and an optional OTHERWISE phrase.

The execution phrase provides ponder with the ability to cause or prevent mental processes, change the options used in resource allocation, or change the active plans. Execution phrases can shut down some aspect of react by canceling all events or pending queue entries associated with a specific target. Ponder can also start up reactions after they have been shut down. Thus ponder can be used to moderate the rate at which reactions occur outside of the normal evaluation rate controls.

Currently, the execution phrase contain 11 options. At least one option must be used when the user employs an execution phrase, and the user is free to select more than one phrase. Table 2.10-4 lists the options and their definitions.

TABLE 2.10-4. Execution Phrase Options.

Option	Description
decentralize cmd-chain	This option causes a message to be built that will change the mode of control specified within a user-defined message to a subordinate type name. Example: Not currently implemented in SWEG instruction files and does not work (obsolete).
invoke plan	This option activates the user-specified plan. If there is an active plan in the plan category, the named plan will replace it; otherwise, the named plan will be initiated in the plan category. The user will specify the name of both the plan and the plan category. Example: INVOKE PLAN just_maneuvering FOR MOVEMENT
employ	This option changes the current active allocation option to the one specified in the option. The user will provide the allocation option and its name. Example: EMPLOY MNVR-START atk_tgt
PENDING QUEUE ADD or DELETE	This option adds or deletes either the specified allocation option or plan category to/from the pending queue at the current game time. The user will select either the add or delete option as well as provide the allocation option or plan category. When plans or allocation options are changed within ponder, the user is responsible for deleting the old allocation options or plan categories from the pending queue. The user will provide either the allocation option or the plan category. Example: PENDING-QUEUE DELETE FOR MOVEMENT PENDING-QUEUE DELETE FOR MNVR-QUEUE-ADD
wake-up	This option schedules a wakeup time for the user-specified allocation option or plan category at the specified time delay from the current relative time. The user will provide the relative time and its unit of measure and either an allocation option or a plan category. Example: WAKE-UP IN 20. (SEC) FOR MOVEMENT
cancel	This option removes the specified allocation option or plan category wakeup event from the event list. If the user is changing an allocation option or a plan category on the event list, the user is responsible for removing the old entry and creating a new one with the WAKEUP option. The user will provide either the allocation option or the plan category. Example: CANCEL MNVR-START CANCEL MOVEMENT
associate	This option is part of the decomposition of orders. Specifically, it permits the user to associate subordinates or platforms to sub-boundaries decomposed by the SPLIT-INTO option. Example: Not currently implemented in SWEG instruction files.

TABLE 2.10-4. Execution Phrase Options. (Contd.)

Option	Description
split-into	This option is also part of the decomposition of orders. It allows the user to decompose lateral boundaries into sub-boundaries and allows ASSOCIATE to assign subordinates or platforms to them. Example: Not currently implemented in SWEG instruction files.
GO-TO LINE or POINT	This option creates an order with the appropriate GO-TO and maneuver instructions. The user will select either a line or a point and specify the geographical control name. The user may provide a maneuver name for transiting to the line or point and specify that the first geographical control point is within another geographical control point. Example: GO-TO POINT enter_orbit_pt CONTAINED-IN GEOGRAPHICAL-CONTROL my_controls MANEUVER racetrack_6
SEND or IMPLEMENT MESSAGE	This option is used to send or implement an order to a resource. The user will select either ALL or THIS: THIS is used to send a message to the current active resource, and ALL is used to send messages to all resources that have movement orders implemented in resource allocation. Orders can be sent to their own platforms . Example: Not currently implemented in SWEG instruction files.
REQUEST EMPLOYMENT-OF or DELIVERY OF	This option builds a message that is sent out by a SEND MESSAGE that requests a resource be delivered to a friendly player or a target. The FILL-REQUEST allocation option processes this request. The user will select either EMPLOYMENT-OF or DELIVERY-OF as this option's key phrase. The user will also specify a resource name, an amount, and a destination. An ASK CDR ON CMD-CHAIN or PLAN is an optional argument. Example: REQUEST EMPLOYMENT-OF ORDNANCE missile-j QUANTITY 5 ROUNDS DESTINATION: PLAYER-TGT ASK CDR ON CMD-CHAIN abn_cp_chain

The following example of a contingency plan for the bvr_fighter contains examples of all three phrases.

```

PLAN go_to_orbit
  FOR-ALL PLATFORM-TYPE
    $ check fuel
    WHEN FUEL-REMAINING < 300. (KG)
      EMPLOY MNVR-START head_home
      FOR-ALL PLAYER-TGT
        PENDING-QUEUE DELETE FOR MNVR-START
        PENDING-QUEUE DELETE FOR MNVR-STOP
        CANCEL MNVR-START
        CANCEL MNVR-STOP
        PENDING-QUEUE ADD FOR MNVR-START
      END-LOOP
    WAKE-UP IN 30. (SEC) FOR MOVEMENT

```

```
        OTHERWISE
        WAKE-UP IN 45. (SEC) FOR MOVEMENT
    END-BRANCH
END-LOOP
END-PLAN
```

The explanation of the example contingency plan will begin with the assumption that the user declared `go_to_orbit` to be the initial plan for movement in the SDB. At the start of the exercise, each player having `go_to_orbit` as its initial plan will think about the plan. The first loop in the plan examines, in turn, each platform belonging to the player. This loop is necessary because the remaining fuel criterion requires a platform to work correctly. If the plan does not include the `PLATFORM-TYPE` loop, the criterion cannot be evaluated effectively. This is one of the major differences between resource allocation and contingency plans. In react there is one target and the examination of each resource is assumed to occur; in ponder these two assumptions are eliminated.

The platform loop is not required for the execution phrase that employs a resource allocation option for maneuver start, and this execution phrase could be moved outside the loop. Its position within the loop causes some unnecessary processing since the resource allocation option will be reset for each iteration of the loop. However, the example player only has one platform, so there is little reason to change the organization of the plan. Making this point illustrates a react assumption in *SWEG*; resource allocation options apply to all resources at once; it is not possible to use one resource allocation option for one resource and then change the option for another resource during the same react decision event.

The next loop examines each perceived platform. This loop is required in order to perform the execution statements. Adding and deleting or canceling resource allocation pending queue entries or events requires a subject perceived platform in order to operate correctly. The two deletions and two cancellations for each target effectively stop the player from thinking about reactive maneuvers. However, if any new targets are detected (directly or indirectly), reactive movement entries will be made to the pending queue. There is no method for causing the player to remember that it does not want to think about reactive maneuvers. Therefore, a contingency plan should be rescheduled periodically to eliminate the reactive maneuver thinking if this is important.

There are two `WAKE-UP` execution statements. The player will think about heading home more often once the remaining fuel falls below 300 kilograms. If the fuel is above this threshold then there is not such a sense of urgency to head home.

2.10.3 Functional Element Software Design

See Section 2.9.3.

2.10.4 Assumptions and Limitations

- Knowledge is the result of one of the following: Initial user instructions, Dynamic updates from sensor results, Dynamic updates from messages, or Implicit updates for self and other `PLAYER`s used as resources.
- Each `PLAYER` with a thinker has its own perceptions, which may be different

from the simulation reality.

- Perceptions are originally derived from model reality.
- Perceptions that are different from model reality are the result of user instructions or time lag since the last update.
- Friendly perceptions are aggregated at the PLAYER level of detail; i.e., unless explicitly detected, the entire PLAYER is perceived to be at the location of one of its PLATFORMs. (Friendly perceptions are those so defined by the user and those which are implicitly obtained whenever a PLAYER evaluates a TACTICAL CRITERION with an associated resource that is not part of the PLAYER. In the second case, kinematic attributes of the resource are implicitly received at the evaluation time; this information is accurate for one of the PLATFORMs of the resource PLAYER.)
- Perceptions of self are implicit, contain all information, and are always accurate.
- Simulated perceptions should be able to differ from simulated reality in terms of detail or aggregation, quantity of information, quality of information, and timeliness; however, in *SWEG*, they may differ only in timeliness.

2.10.5 Known Problems or Anomalies

None.

